File permission

u	owner of file/directory (user)
g	group of the file/directory
o	other
a	all

Permission Types:

permission	octal value	Meaning
Read (r)	4	The file can is read only, for directory it's contain can be listed
Write (w)	2	File can be modified, for directory we can create, remove files/directory
Execute (e)	1	Files can be executed if it is a program file, change into directory (cd)

Note:
No-Permission (-) o

Permission operator

+	Add permission
-	remove permission
=	To assign (absolutely) permission

#ls -l
_ _ _ _ _ _ _ _ _ _ in the left part of output show the permission.

1st position indicates the type o file

-	Normal
d	directory
l	linkfile
b	block device file
c	character device file

- 2nd, 3rd and 4th positon indicates the permission for user
- 5th, 6th and 7th position indicates the permission for group
- 8th, 9th and 10th position indicates the permission for other

For example, when a new file is created it's permission will be as under:
-rw-r--r--: meaning that, the file has read/write permission for user, and read permission for group and others.

Deafult file permission
umask: It is used to set default permission on file/directory on its creation.
Maximum allowed permission on file (666) and Maximum allower permission on directory is (777)

Example:
Set the value of umask such that permsiion on a file during it's creation give read/write access to the

owner, read permission to group and no permission to other.

```
#umask 026
```

Formula to calculate umask:
for directory subtract the file permission value from 777.
for file subtract the file permission value from 666.
In above exampe: file permission for owner is read(4)/write(2), sum value is (4+2=6), file permission for group is read (4), and permission for other is 0

```
 666
-026
 640
```

Note: in case the above file is a directory, we should use 777 in place of 666

The default value is set in */etc/bashrc* file.

Changing permission of created files/directories
chmod [option] [mode/permission] <file/directory>

Examples:
write a command to assign file permission as under:

for owner:	full (read/write/execute)
for group:	read/write
for other:	read only

```
#chmod u=rwx,g=rw,o=r myfile.txt
or
#chmod 764 myfile.txt
```

Write a command to assign file permission as under:

for owner:	full
for group:	read/execute
for other:	none

```
#chmod u=rwx,g=rx,o= myfile.txt
or
#chmod 750    myfile.txt
```

Write a command to assign file permission as under:

for owner:	read
for group:	none
for other:	none

```
#chmod u=r,g-rwx,o-rwx,o-rwx myfile.txt
#chmod 400 myfile.txt
```

Again, change the file permission to execute for all

```
#chmod a+x myfile.txt
```

Write a command to assign directory permission (also to it's content) as under:

for owner:	full
for group:	none
for other:	none

```
#chmod -R 700 mydirectory
```

Working with files,I/o, pipe,process and text processing

locate

```
$locate myfile
```

updatedb : update system database.

Find

Find [path] [options] [action]

find all the files whose name start with s (case insensetive)

```
    #find . -iname “s*”
```

find all the files whose name start with s (case sensetive)

```
    #find . -name “s*”
```

Find all the files in the system that are norm files and delongs to the user1

```
    #find / -user user1 -and -type f
```

Locate all files (not directories) with user user1 and (-and operator) type f (file)

d - directory
c - character
b - block
l - link file

Find all the empty files in you system (current directory location)

```
    #find . -size 0c
```

Find all the files those are 10 character long

```
    #find . -size 10c
```

find all the files those are more than 100c long

```
    #find . -size +100c
```

find all the empty files in your current directory and remove these files.

```
    #find -size 0 -exec rm { } \;
    #find -size 0 -ok rm { } \;
```

\; is a escape character which will output to ;
Here ok ask for conformation before next step
exec does not ask for conformation -exec rm { } \;

{ } flace holders for found files

String Processing

Word count wc:

```
    #wc /etc/passwd
```

shows number of lines, word, and characters for /etc/passwd

```
    #wc -l /etc/passwd
```

shows number of lines in the file

```
    #wc /etc/passwd /etc/group
```

shows number of lines, word, and characters of two files and also shows grand total.

sort:

```
    #sort -n -k 3 -t : /etc/passwd
```

Options

```
        -n numeric sort
        -k field
        -t seperator
        -r reverse sort
        -f ignore case
        -u unique sorting (if duplicate record found only on line is shown)
```

cut

Display the list of all the users in your system.

```
        #cut -f 1 -d : /etc/passwd
```

Display the list of all the users in your system. Also display it's uid and home directory.

```
        #cut -f1,3,6 -d: /etc/passwd
```

Display first four fields in /etc/passwd

```
        #cut -f-4 -d: /etc/passwd
        or
        #cut -f1-4 -d: /etc/passwd
```

Display 4th to 7th fields of /etc/passwd

```
        #cut -f4- -d: /etc/passwd
        or
        #cut -f4-7 -d: /etc/passwd
```

Display 1st three character in file /etc/passwd

```
        #cut -c-3 /etc/passwd
```

Cuts characters except beginning three characters from all line.

```
        #cut -c3- /etc/passwd
```

Determine the types of shells used by different users in you system

```
        #cut -f1,7 -d: /etc/passwd
```

paste

For our example please make these files with content as below

file: alpha contains	numeric contains
a	1
b	2
c	3
d	4

```
#paste alpha numeric
```

output:

```
a       1
b       2
c       3
d       4
```

```
#paste -d: alpha numeric
```

Output

```
a:1
b:2
c:3
d:4
```

tr

Note: Translate, squeeze, and/or delete characters from standard input,
 writing to standard output.

the following command translate all characters from a-m int capital A-M

```
#cat /etc/passwd | tr 'a-m'A-M'
```

conver all a as b in file listing. (Do not effect in actual file)

```
#cut -f1 -d: /etc/passwd | tr 'a' 'b'
```

diff

```
#diff myfile.old myfile.new
```

aspell: checks the spelling

```
#aspell -l en check m.txt
```

standard I/O and pipes

redirection operator

>	overwrite
>>	append

pipe

Display the shell and number of user for each shell

```
#cut -f7 -d: /etc/passwd|sort|uniq -c
```

the output of cut command is passed as input for sort, the output of sort is passed to uniq command. -c option in uniq cable is use for count.

Process

software program in execution is called process.
Each process is identified by a process Identification number (PID)
PID 1 is assigned to init, which is the first process that stands at boot time.

```
#pstree
#ps
```

Process Status

R	runnable
S	sleeping
T	stopped
D	uninterruptable sleep
Z	zombic
N	low priority process
<	high priority process
w	No resident pages in the memory

Sending Signals to processes

TERM(15) soft signal
KILL(9) strong signal

```
#kill -TERM <pid>
#kill -15 <pid>
#kill <pid>
```

Terminating process

```
        Normal end
        Ctrl + c
        kill -TERM <PID>
        kill -9 <PID>
```

Altering Process scheduling priority

Max -20
Min 19
default 0

nice

```
        #nice -n -10 find /
```

renice

```
        #renice -n 11 init
```

To run the process in background use & sign at the end

```
#find / >output.txt &
[1] 7689
```

1 is the job id and 7689 is the pid.

to view the background processes use jobs

```
#jobs
```

Stopping/suspending a process

```
ctrl+z
```

Resuming the stopped process

running resumed process in background

```
#bg %<jobid>
```

running resumed process in forground

```
#fg %<jobid>
```

Text processing

grep command

Determine whether a user shiba exist in the system or no

```
#grep 'shiba' /etc/passwd
```

case when u have to show only username

```
#cut -f1 -d: /etc/passwd|grep 'shiba'
```

Display the list of all users in system that uses bash in end of line

```
#grep bash$ /etc/passwd
```

Display the list of all users in system that does use bash in end of line

```
#grep -v bash$ /etc/passwd
```

Display all the files that contain shiba in it. The files should be located in /etc/ directory

```
#grep -l shiba /etc/*
```

Display the list of directories only in your current directory.

```
# ls -l  | grep ^d
```

In the above ^ indicates beginning of line. The if the beginning of line contains d it will display it.

Display the lines with line number that contain cat/Cat word in the file myfile.txt

```
#grep [Cc]at myfile.txt
```

[grep options]

-n	show line number
-c	count
-l <text>	list file with content shiba
-R	recursive, also searches in sub directories

sed

show all words cat in the file myfile.txt as dog.

```
#sed -e 's/cat/dog/g' m.txt
```

show all words cat in the file myfile.txt as dog if the cat word is at the beginning of line.

```
#sed -e 's/^cat/dog/g' m.txt
```

show all words cat in the file myfile.txt as dog if the cat word is at the end of line.

```
#sed -e 's/cat$/dog/g' m.txt
```

awk

```
awk pattern {action}
```

Write an awk statement to find the list of all user that use bash as the shell.

```
#awk '/bash/{print}' /etc/passwd
```

To print all the content.

```
awk '{print}' /etc/passwd
```

To print field 1(user) and field 6(home directories)

```
#awk -F: '{print $1,$6}' /etc/passwd
```

To print field 1(user) and field 6(home directories)

```
# awk -F: '{print "The home directory of user " $1 is " " $6}' /etc/passwd
```

output will be similar to following

```
The home directory of user cba /home/cba
```

To print “shiba is the actual spelling for cba” if the user cba exist in the file /etc/passwd

```
#awk -F: '{if ($1 ~"cba") print "shiba is a actual spelling for " $1 }' /etc/passwd
```

To print the sum of all the values in 3rd field

```
#awk -F: '{ sum += $3; } END { print sum; }' /etc/passwd
```